Diseño de un componente de simulación para comunicaciones en redes de sensores

Pi Puig Martín[1], Medina Santiago[1], Batista Ary[2], Encinas Diego[1], Romero Fernando[1], De Giusti Armando[1,3], Tinetti Fernando G.[1,4]

[1]Instituto de Investigación en Informática LIDI (III-LIDI), Facultad de Informática, Universidad Nacional de La Plata, 50 y 120 2do piso, La Plata, Argentina.
[2]Universidad Nacional de Quilmes, Roque Sáenz Peña 352, Bernal, Argentina.
[3]CONICET – Consejo Nacional de Investigaciones Científicas y Técnicas
[4]CIC – Comisión de Investigaciones de la Pcia. de Buenos Aires
{mpipuig, smedina}@lidi.info.unlp.edu.ar, ary.batista@unq.edu.ar, {dencinas, fromero, degiusti, fernando}@lidi.info.unlp.edu.ar

Abstract. Los Sistemas Distribuidos de Tiempo Real deben ejecutar algoritmos en plazos de tiempo adecuados a los requerimientos de la implementación. Durante la etapa de verificación y validación del hardware, pueden encontrarse medidas que impliquen la realización de cambios o modificaciones en los mismos. Una manera de disminuir la complejidad y la probabilidad de errores en la generación de hardware es desarrollar una simulación específica de éstos. Se propone el diseño, implementación y validación de un modelo de simulación de un sistema de comunicaciones Controller Area Network, utilizando el entorno de simulación Proteus. La finalidad del modelo es predecir el comportamiento de la transmisión en diferentes escenarios.

Keywords: Modelado, Simulación, Comunicaciones en Sistemas de Tiempo Real, Controller Area Network.

1 Introducción

La infraestructura de comunicación en un sistema de tiempo real distribuido debe cumplir con requerimientos no funcionales [15] muy diferentes a los demandados por otros sistemas. Estos requerimientos son: exactitud, fiabilidad, flexibilidad y estructural.

En el requerimiento de exactitud se puede notar la necesidad de conseguir una latencia corta y un jitter mínimo, siendo esta la principal diferencia entre los sistemas de comunicaciones de tiempo real y los otros. La latencia tiene una duración que inicia con la lectura de los sensores en cada uno de los nodos y finaliza con la salida del actuador correspondiente, teniendo en cuenta el procesamiento requerido. El jitter es la diferencia entre las latencias del peor y mejor caso, y un valor alto de este parámetro puede producir efectos negativos para calcular el instante de aceptación de un mensaje.

La fiabilidad engloba características que necesitan poseer los sistemas de comunicaciones para aportar seguridad a la comunicación (esto se consigue por medio de redundancia). También, un protocolo de comunicaciones de tiempo real debe ser lo suficientemente flexible para soportar cambios sin la necesidad de requerir modificaciones en el software. Los requerimientos estructurales de la red están determinados no solo por consideraciones técnicas (requerimientos funcionales) sino también económicas.

Uno de los primeros pasos en la fase de diseño de estos sistemas es efectuar un análisis de funcionamiento de los mismos. En la actualidad esto se consigue por medio de técnicas de simulación; de esta manera se permite estudiar, entre otros aspectos:

1) la interacción entre los distintos componentes
2) protocolos de comunicaciones
3) modos de operación
4) potencia consumida de los distintos subsistemas.

Debe quedar claro que el propósito de la simulación funcional no es representar el funcionamiento exacto de los componentes ya que se perdería la ventaja de abstracción de las características físicas [1].

En este trabajo se propone un modelo de simulación de un dispositivo de comunicaciones CAN [2], utilizando el entorno de simulación Proteus [3] que actualmente no ofrece este protocolo. Luego la validación del modelo se realizará por medio de un sistema físico con el cual se contrastarán los datos obtenidos con las pruebas en el simulador.

El trabajo está organizado de la siguiente forma: en la segunda sección se describirán algunas características del protocolo de comunicaciones CAN. En la tercera sección se explica la metodología utilizada para desarrollar el modelo del sistema. En la cuarta sección se muestran el desarrollo de la simulación. En la quinta sección se explica la validación y los resultados obtenidos. Finalmente, en la sexta sección las conclusiones y trabajos futuros.

2 Protocolo CAN

Para este trabajo se ha elegido el protocolo CAN el cual ha sido formalmente definido por el estándar ISO 11898 [4]. Este protocolo fue diseñado por la compañía Robert Bosch GmbH en Alemania y el propósito inicial era crear una red más rápida y resistente a interferencias dentro del ámbito automotriz. El estándar ofrece una comunicación determinística, con prioridades y detección de errores. La topología del bus consiste generalmente en un medio de transmisión, formado por un par de cables, y estaciones o nodos. Aunque la especificación del medio de transmisión del cable no está indicada en la norma original puede ser adaptada dependiendo el área de aplicación. La codificación utilizada es de Non Return to Zero (NRZ) con stuffing sobre un canal de señales diferenciales (balanceado) lo que asegura mensajes compactos con un mínimo número de transiciones y una alta resistencia a las

perturbaciones externas. En la Figura 1 puede verse el conexionado típico de los nodos a la red y los elementos que la componen.

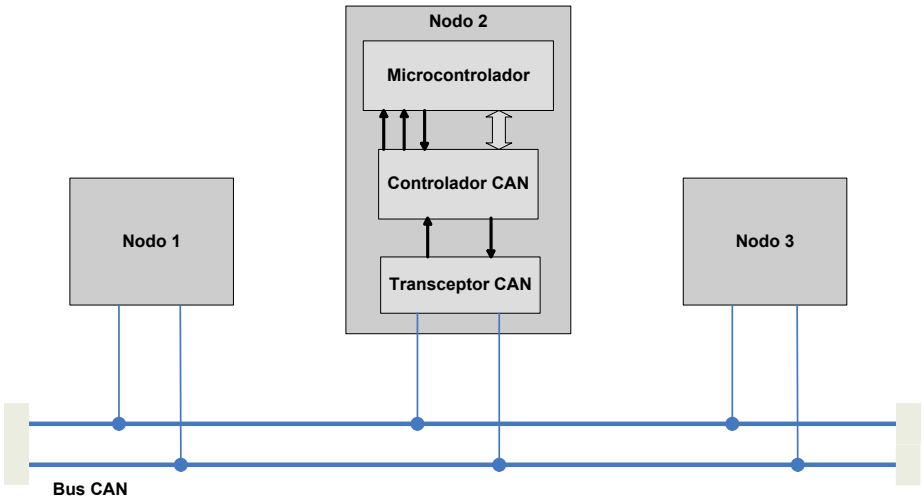

Fig. 1.

CAN es conocido como un protocolo de comunicaciones serie de alta confiabilidad, robustez y performance; además es apropiado, mediante un desarrollo de capa de aplicación, para el control de sistemas distribuidos de tiempo real. Sus principales características son:

- Priorización de mensajes.
- Sistema Multi-maestro.
- Configuración flexible.
- Velocidad de transmisión media (hasta 1 Mbit/s).
- Señalización y detección de fallas.

El mecanismo de transacción usado por el protocolo CAN para transferir datos es similar al modelo productor consumidor en donde cada nodo envía mensajes sin requerir confirmación y de la forma multicast o broadcast. El tipo de acceso al bus CAN es multi-maestro; por este motivo, el derecho a acceder al bus no es asignado por un nodo central pero de todas formas todos los nodos pueden intentar acceder al mismo tiempo. El arbitraje se resuelve por un procedimiento que hace uso de los identificadores que se encuentran al mismo tiempo en el bus.

La transferencia de mensajes se manifiesta y controla por cuatro tipos de trama diferentes: tramas de datos, tramas remotas, tramas de error y tramas de sobrecarga. En la Figura 2 se puede observar los campos de la trama de datos.

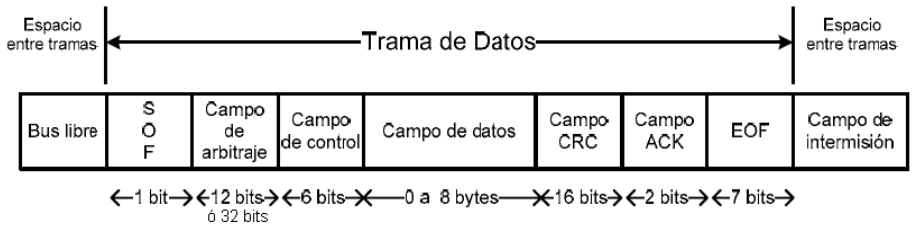

Fig. 2.

3 Metodología

3.1 Emisión de mensajes

El modelo de simulación se realiza considerando un análisis de planificación de transmisión de mensajes para redes CAN pero imponiendo restricciones propias del desarrollo de un simulador.

El modelo de planificación CAN utilizado es el propuesto en [5] y en sus trabajos previos. Un concepto de planificación importante es que cada mensaje es encolado por una tarea, proceso o interrupción software que se ejecuta en el procesador host. Esta tarea es invocada por un evento o consulta de estado con una cantidad de tiempo limitada para encolar el mensaje que esté listo para transmitir. Este tiempo varía entre 0 y Jm, donde Jm es denominado el jitter de encolamiento del mensaje.

Generalmente los modelos de planificación de prioridad fija como es el caso del protocolo CAN, presentan jitters a pesar de poseer sólo tareas periódicas [6]. Existen varios trabajos que realizan un análisis de planificabilidad del protocolo [7] [8] [9] e introducen al jitter como un parámetro necesario para efectuar la planificación de mensajes que se envían periódicamente.

La presencia de un jitter aleatorio provoca un retardo del tiempo de transmisión, en [10] se realiza un análisis estocástico de los tiempos de respuesta y se propone una función de distribución uniforme para el jitter. Pero si se considera que el jitter de un componente está formado por la contribución de distintos subcomponentes, que poseen a su vez otros jitters, puede apelarse al teorema del límite central. El teorema del límite central establece que la función distribución de probabilidad de la suma de un número suficientemente grande de variables aleatorias, con distintas distribuciones, puede aproximarse a una normal o gaussiana [11]. Esta es una de las razones, para proponer como función de distribución de probabilidad del jitter a la normal o gaussiana.

3.2 Ventanas de lectura y escritura

El envío y recepción de mensajes está implementado por medio del concepto de ventanas de escritura y lectura. Las ventanas están diseñadas para tener un intervalo de tiempo fijo y representar las restricciones que poseen los mensajes cuando llegan

en un momento o periodo en el que no pueden transmitir a pesar de poseer una alta prioridad. Por lo tanto, cuando un mensaje llega durante un intervalo de tiempo de ventana de lectura debe esperar hasta el inicio de ventana de escritura para competir con otros posibles mensajes en el momento de arbitraje.

3.3 Arbitraje

Cada nodo escribe el bit correspondiente a su identificador durante el periodo de ventana de escritura y luego, en la ventana de lectura, se verifica que el bit presente en el bus corresponde al escrito. Sí este bit difiere, significa que existe un mensaje emitido por un nodo con mayor prioridad. Entonces los nodos con menor prioridad se dedican a recibir el mensaje que se está transmitiendo en el canal.

Es decir, un nodo logra atravesar el proceso de arbitraje si puede escribir por completo su identificador. De esta forma obtiene el control del bus y procede a emitir su mensaje.

3.4 Reloj global

La sincronización entre los nodos de la red está dada por una señal provista por el entorno de simulación. Esta señal se emite a intervalos regulares sobre un canal de sincronización y determina el inicio y fin de las ventanas de escritura y lectura.

Por lo tanto, se asume un reloj global suponiendo que los métodos Hard Synchronization y Resynchronization [12] son suficientes para sincronizar las señales de reloj de cada nodo en el sistema real.

3.5 Herramientas utilizadas

Para el modelado y la implementación de la simulación, se utilizó un conjunto de herramientas, las cuales se listan a continuación.

- Proteus v7.7
- CCS Compiler v5.0

Luego, dentro de la suite de simulación Proteus, se decidió utilizar microcontroladores del tipo PIC16F877A como componentes básicos de implementación de los nodos y controladores CAN.

Para la implementación del sistema físico, el microcontrolador seleccionado fue el AT89S52, el controlador CAN elegido fue el SJ1000 y el transceptor CAN el componente PCA82C250.

4 Desarrollo

4.1 Implementación

En el sistema básicamente existen dos módulos principales: el Nodo y el Bus. Como sucede físicamente, el Nodo modelado está compuesto por un microcontrolador y un controlador CAN. En este modelo de simulación no se ha tenido en cuenta al transceptor CAN ya que este componente agrega un retardo fijo del orden de los nanosegundos, el cual es un valor pequeño en comparación al tiempo necesario para la transmisión de un bit (1 μseg).
En la Figura 3 se muestra el diagrama de comportamiento de la implementación para el envío de un paquete de datos CAN.

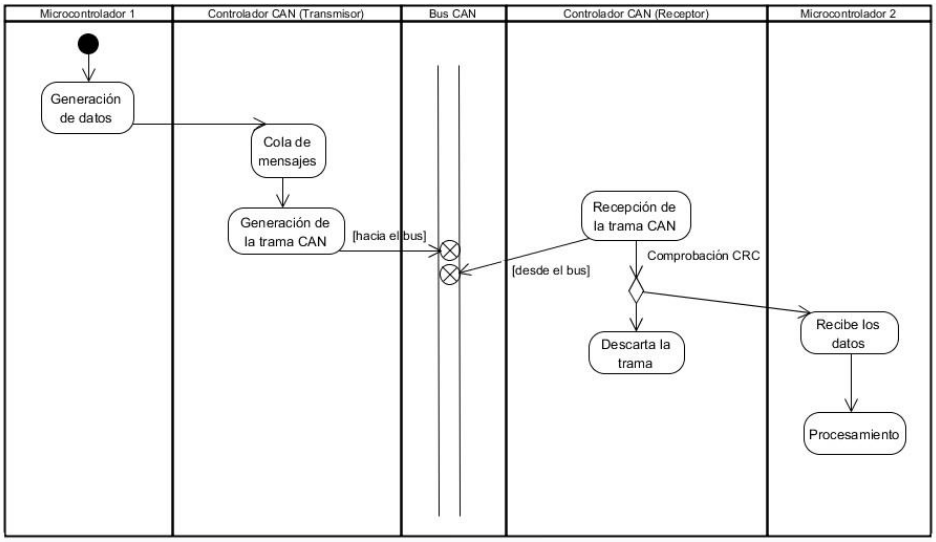

Fig. 3.

4.2 Modelo de programación

En la Figura 4 se muestra un diagrama de actividades de los componentes principales del sistema de simulación.

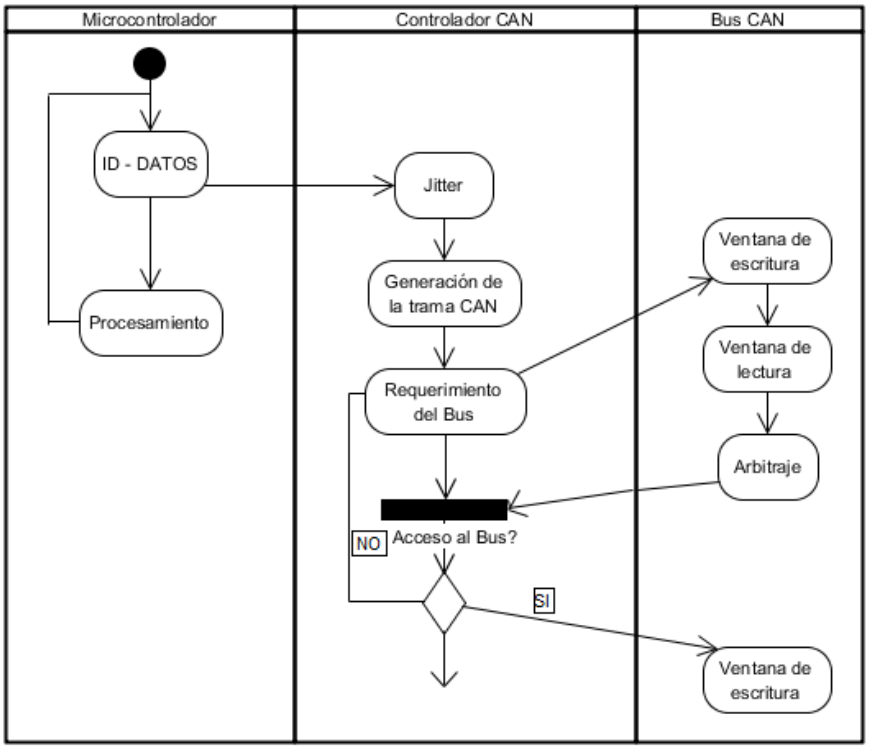

Fig. 4.

El microcontrolador cumple las funciones de sensor del sistema y de host del controlador CAN. En este componente se generan el dato e identificador del mensaje. Estos parámetros se transmiten al controlador CAN por medio del protocolo RS-232.

El controlador CAN junto con el bus, es uno de los principales componentes del sistema de simulación. Por medio de diferentes funciones se completan los campos de la trama de datos, se introduce el dato a transmitir y se agrega el jitter correspondiente al buffer de transmisión. Como se ha mencionado en la sección 3.1 se propone representar al jitter con una función de distribución normal. Se implementó una función que aplica la transformada de Box-Muller [13] para la generación de números aleatorios con la función de densidad de probabilidad normal o gaussiana.

El bus de datos (Bus CAN) se realizó utilizando compuertas lógicas del tipo OR para permitir a cada nodo incrementar el valor de la ventana actual. Y una compuerta del tipo AND que reinicia el bus cada vez que el reloj global vuelve a cero el canal de sincronización.

5 Resultados

5.1 Métricas obtenidas con el sistema físico

Seleccionando una periodicidad de envío de 4 mseg, la tasa de transmisión programada fue de 3,96 mseg. Esto es consecuencia de la arquitectura del microcontrolador (tamaño de registros), el proceso de transmisión del controlador CAN y la frecuencia de los cristales en cada nodo.

En general, el tráfico en redes de datos tradicionales se analiza a partir del throughput, latencia y jitter [14] de la red. Aunque en una red para sistemas de tiempo real estricto estos parámetros son necesarios para cuantificar la calidad de servicio (QoS), por las características específicas de las redes en el protocolo CAN medir el throughput es poco significativo. De hecho el throughput es fijo y de 1 Mbps, ya que es la máxima velocidad de transferencia que admite el protocolo. Es por esto que el parámetro que se evalúa es el Inter Arrivals Time (IAT) de mensajes.

La cantidad de mensajes enviados fue de 37339 aproximadamente en 150 seg (tasa de 3,96 mseg). En la Figura 5 se puede ver las métricas obtenidas.

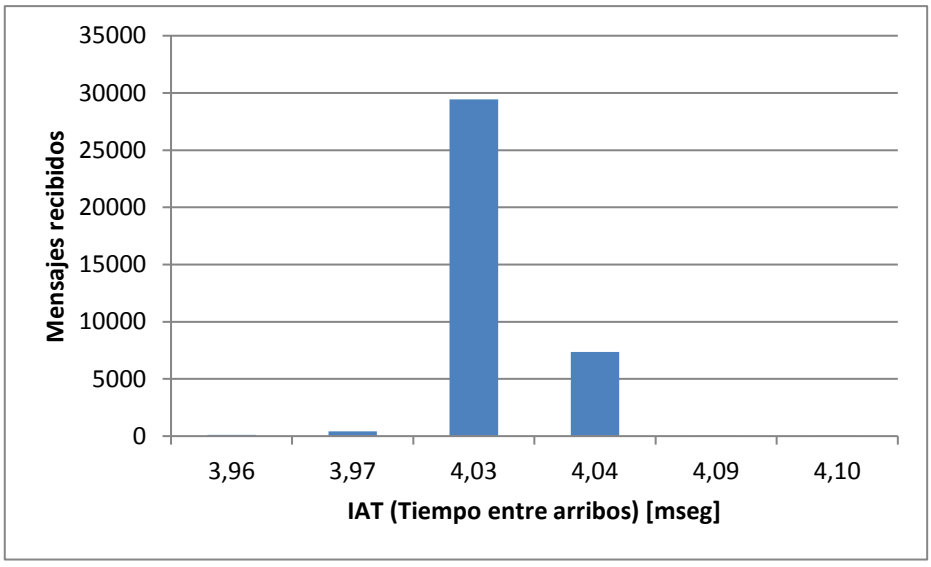

Fig. 5.

Como puede observarse en la Figura 5, el mayor porcentaje de mensajes se recibe a un IAT promedio de 4,03 mseg y el resto de los mensajes no se desvían más de 70 µseg de ese valor.

5.2 Métricas obtenidas con el modelo de simulación

El modelo de simulación agrega un overhead en la ejecución comprendido por: tiempos de procesamiento para implementar el CRC del protocolo CAN, arbitraje (por medio de ventanas de lectura y escritura) y el jitter de distribución gaussiana.

El escenario que se simuló fue realizado en base al sistema físico, un nodo que transmite a una tasa periódica de 3,96 mseg un mensaje de 8 bytes de datos a un nodo receptor. Nuevamente la prueba consistió en enviar 37339 mensajes. En la Figura 6 pueden verse las métricas obtenidas.

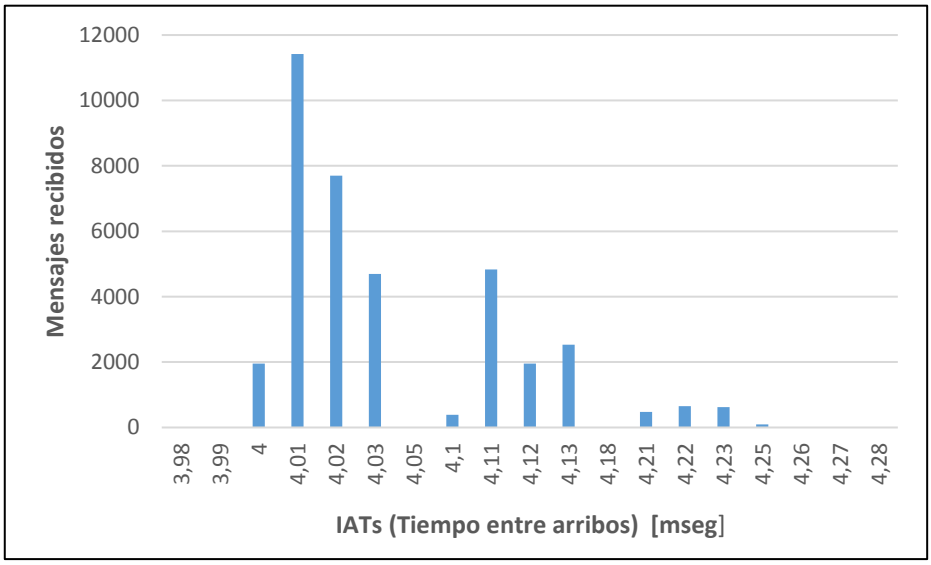

Fig. 6.

En la Figura 6 puede verse como principal diferencia con el sistema físico, que el valor del IAT promedio es de 4,05 mseg con una desviación máxima de 230 µseg. Pero por otra parte aumenta el número de IATs, permitiendo obtener una mayor resolución. Aun así puede observarse un comportamiento similar en los dos modelos.

6 Conclusiones

De la observación de los resultados del modelo de simulación se puede decir que cumple con la especificación del protocolo CAN y de esta forma es validado. Además, de la comparación de los resultados de la ejecución de simulación con los obtenidos con el sistema físico se puede afirmar que con los mismos datos de entrada los dos sistemas producen salidas similares.

Desarrollar la implementación necesaria para que el modelo de simulación soporte otros tipos de tramas CAN, mejorar el escalamiento del bus y analizar otras

distribuciones de probabilidades para el jitter, serían los próximos pasos para proseguir con el presente trabajo.

Bibliografía

[1] EICKHOFF J. (2009). Simulating Spacecraft Systems. Springer.

[2] Robert Bosch GmbH. CAN Specification 2.0. 1991

[3] Proteus. https://www.labcenter.com. 2016

[4] ISO 11898: Road Vehicles – Interchange of digital information – Controller Area Network (CAN) for high speed communication. 1993.

[5] R. Davis, «Controller Area Network (CAN) schedulability analysis: Refuted, revisited and revised.,» Real-Time Systems. Springer, vol. 35, nº 3, pp. 239-272, 2007.

[6] A. Burns, Real-time systems and programming languages, Addison Wesley, 2009.

[7] R. Davis, «Controller Area Network (CAN) Schedulability analysis with FIFO queues,» 23rd Euromicro conference on Real-Time Systems, pp. 45-56, 2011.

[8] P. Yomsi, «Controller Area Network (CAN): Response time analysis with offsets,» 9th IEEE International workshop on Factory Communication Systems, pp. 43-52, 2012.

[9] N. Navet, «Controller Area Network (CAN) Schedulability analysis for messages with arbitrary deadlines in FIFO and work-conserving queues,» 9th IEEE International workshop on Factory Communication Systems, pp. 33-42, 2012.

[10] M. Di Natale, Undertanding and Using the Controller Area Network Communication Protocol. Theory and Practice, Springer, 2012.

[11] J. Devore, Probabilidad y estadística para ingeniería y ciencias. Sexta Edición, Thomson Learning, 2005.

[12] Philips semiconductors, Application note. Determination of Bit Timing Parameters for the CAN Controller SJA 1000. AN97046, 1997.

[13] Box, G. E. P.; Muller, Mervin E. A Note on the Generation of Random Normal Deviates. Ann. Math. Statist. 29. 1958

[14] J. Kurose, Redes de Computadores. 2° Edición, Pearson-Addison Wesley, 2004

[15] Hermann Kopetz.Real-Time Systems.Design Principles for Distributed Embedded Applications. Second Edition. Springer. 2011.ISSN 1867-321X e-ISSN 1867-3228 ISBN 978-1-4419-8236-0 e-ISBN 978-1-4419-8237-7